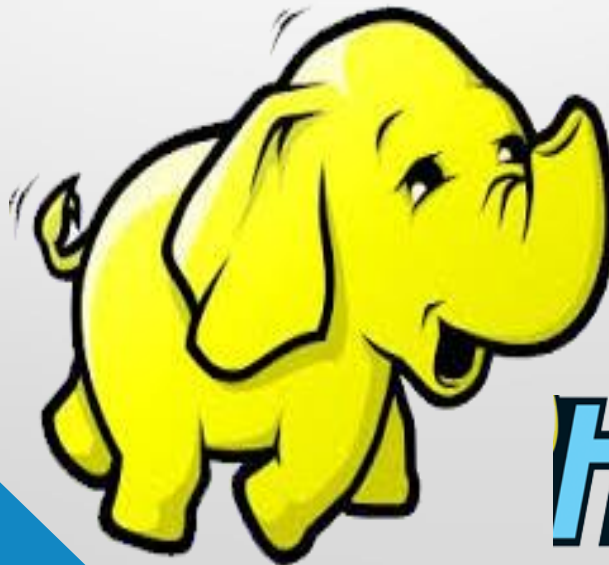


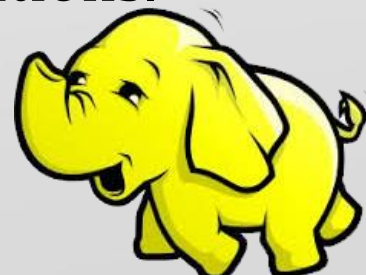
Hadoop, a distributed framework for Big Data



hadoop

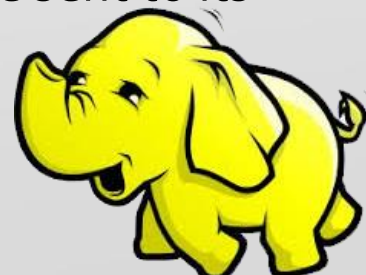
What is Big Data?

- Big data is about storing, processing huge volume of data with the ultimate goal of generating insights from the data.
- Data can be processed real time or in batches depending on the solution requirements.
- Traditional database systems like RDBMS are not designed to store, process and analyze huge volume of data.
- Big data is a combination of unstructured, semi-structured or structured data collected by organizations. This data can be mined to gain insights and used in machine learning projects, predictive modeling and other advanced analytics applications.



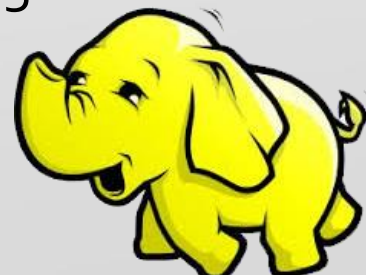
Big Data 5V's Characteristics

- **Volume:** It refers to the amount of data that exists. Volume is like the base of big data, as it is the initial size and amount of data that is collected. If the volume of data is large enough, it can be considered big data. What is considered to be big data is relative, though, and will change depending on the available computing power that's on the market.
- **Velocity:** The speed at which the data enters the solution, higher speed means lesser time to analyze so any solution we build must be able to process the data in acceptable time frame. This is an important aspect for companies need that need their data to flow quickly, so it's available at the right times to make the best business decisions possible. An organization that uses big data will have a large and continuous flow of data that is being created and sent to its end destination. Data could flow from sources such as machines, networks, smartphones or social media. This data needs to be digested and analyzed quickly, and sometimes in near real time. As an example, in healthcare, there are many medical devices made today to monitor patients and collect data. From in-hospital medical equipment to wearable devices, collected data needs to be sent to its destination and analyzed quickly.



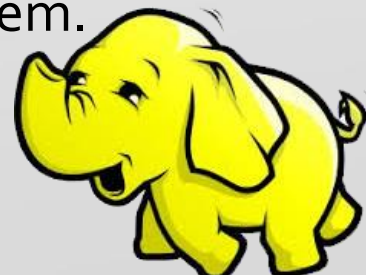
Big Data 5V's Characteristics

- **Variety:** Variety refers to the diversity of data types. An organization might obtain data from a number of different data sources, which may vary in value. Data can come from sources in and outside an enterprise as well. The challenge in variety concerns the standardization and distribution of all data being collected. Collected data can be unstructured, semi-structured or structured in nature. Unstructured data is data that is unorganized and comes in different files or formats. Typically, unstructured data is not a good fit for a mainstream relational database because it doesn't fit into conventional data models. This includes text, images and videos. Semi-structured data is data that has not been organized into a specialized repository but has associated information, such as metadata. This makes it easier to process than unstructured data. This also includes key-value pairs that are grouped into elements within a file. Example : Data stored in XML or JSON files. Structured data, meanwhile, is data that has been organized into a formatted repository.



Big Data 5V's Characteristics

- **Veracity:** It refers to the degree of quality and accuracy of data. Gathered data could have missing pieces, may be inaccurate or may not be able to provide real, valuable insight. Veracity, overall, refers to the level of trust there is in the collected data. Data can sometimes become messy and difficult to use. A large amount of data can cause more confusion than insights if it's incomplete. For example, concerning the medical field, if data about what drugs a patient is taking is incomplete, then the patient's life may be endangered.
- **Value:** The data is of no use unless it adds value to the organization process. It relates directly to what organizations can do with that collected data. Being able to pull value from big data is a requirement, as the value of big data increases significantly depending on the insights that can be gained from them.



Modern Data Analytical Tools

- R and Python
- Microsoft Excel
- Tableau
- RapidMiner
- KNIME
- Power BI
- Apache Spark
- Pig
- Hive
- MongoDB


Difference between reporting and analysis

The ultimate goal for the reporting and analysis is to increase sales and reduce costs.

Both reporting and analysis play roles in influencing and driving the actions which lead to greater value in organizations.

Reporting translates raw data into information. Reporting basically answers ; What is happening?

Analysis transforms data and information into insights. It answers the questions; Why it is happening and what you should do for it.



Challenges of Conventional System for Big Data Leading to use of Modern Technologies like Hadoop

- **Data integration:** Normally, an organization will connect data from numerous sources, which makes it hard to monitor the effectiveness of the integration process. A lot of the problems pertaining to inaccurate insights may be tracked back all the way to how the data gets collected, verified, stored, and utilized. The problem is, when working in data-sensitive and intensive industries, even the smallest error may prove fatal to the success of the overall process.
- **Data storage and complexity:** It is a known fact that the data flowing in is getting more complicated on a daily basis. This means that the systems have to account for a lot more factors and parameters than earlier. As the lot of raw data flows in from various sources, such as salespeople, operations, consumers, and other users existing within the same organization, it becomes evident that the data structure is more complex than ever before. Moreover, there are lots of technologies that function across multiple channels, from offline to web to mobile, which intensifies the complexity to a whole new level and increases the challenges.


- **Data security:** Merely collecting and storing lot of information from different sources isn't enough. The available data needs to be secured at all costs. The creation of technologies such as the cloud helps store data on a single platform so that it can be accessed anywhere, anytime, and make life easier for organizations and consumers alike.
- **Data capture:** Today, the new data is captured directly at the source, which might exist underneath the ocean as seen in gas and oil exploration, or orbiting satellites as evident from weather applications. Example: it is captured on your phone every time you capture a video or image, or send out a tweet. The conventional system does not allow to do data capturing instantly.

Challenges of conventional systems for Big data

- Big data is the storage and analysis of large data sets.
- These are complex data sets that can be both structured or unstructured.
- They are so large that it is not possible to work on them with traditional analytical tools.
- One of the major challenges of conventional systems was the uncertainty of the Data Management Landscape.
- Big data is continuously expanding, there are new companies and technologies that are being developed every day.
- These days, organizations are realising the value they get out of big data analytics and hence they are deploying big data tools and processes to bring more efficiency in their work environment.

Big Data Analytics for Business.

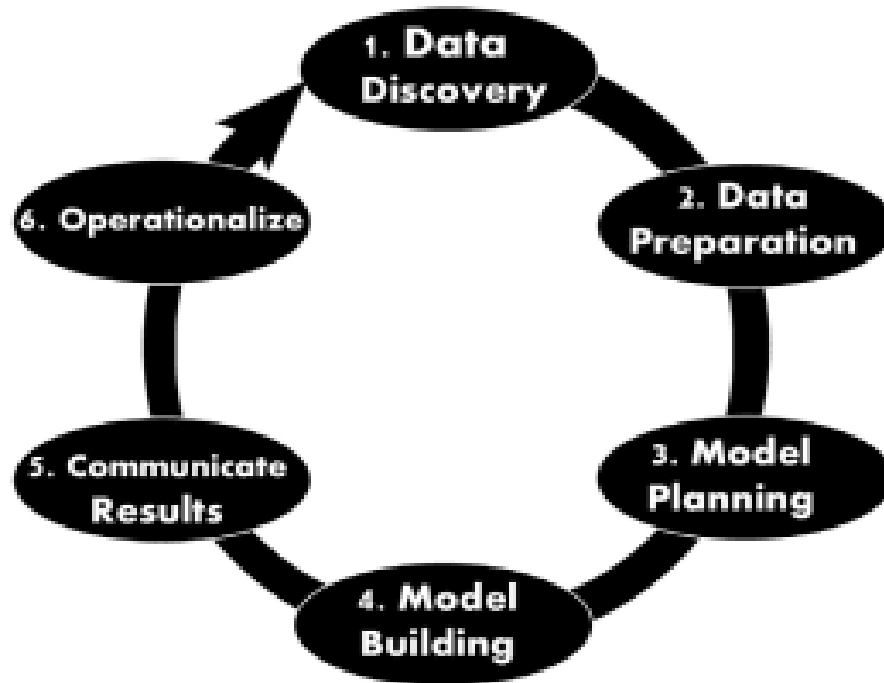
- Big data is used by several multinational companies to filter the data and business of various companies. Companies are taking advantage of big data . It can benefit companies to make adequate decisions in less time and make their work more easier and profitable.
- The major role of big data in any company is to make better business decisions. It can enhance internal efficiency and operations for nearly any type of business.
- It is used by different organizations to harness their data and use it to identify new opportunities. This leads to smarter business moves, more efficient operations , higher profits and happier customers.
- Modern big data analytics and operations anticipate the patterns of consumers. These patterns are then used to motivate brand loyalty as they can collect more data to observe more trends by delivering smarter services and products.
- Big data plays vital role in health care industry, sports industry , insurance industry, banking and securities, retail and wholesale trade,
- Amazon, American express, Netflix, miniclip, general electric, yahoo, facebook, star bucks , yahoo are some of the companies using big data for business.



Big data can present an abundance of new growth opportunities, from internal insights to front-facing customer interactions. Three major business opportunities include: automation, in-depth insights, and data-driven decision making.

- **Automation:** Big data has the potential to improve internal efficiencies and operations through robotic process automation. Huge amounts of real-time data can be immediately analyzed and built into business processes for automated decision making. With scalable IT infrastructure and decreasing cloud computing costs, automating data collection and storage is within reach.
- **In-depth insights:** Big data can also be used to discover hidden opportunities that were unknown to organizations before the ability to review large sets of data. Complex data sets can even be used to develop new products or enhance existing ones. Proprietary data within the market can prove invaluable in the competitive landscape.
- **Faster, better decision making:** With the speed of data analytics technology, paired with the ability to analyze new sources of data, businesses are now able to analyze information instantly and make smart, informed decisions.

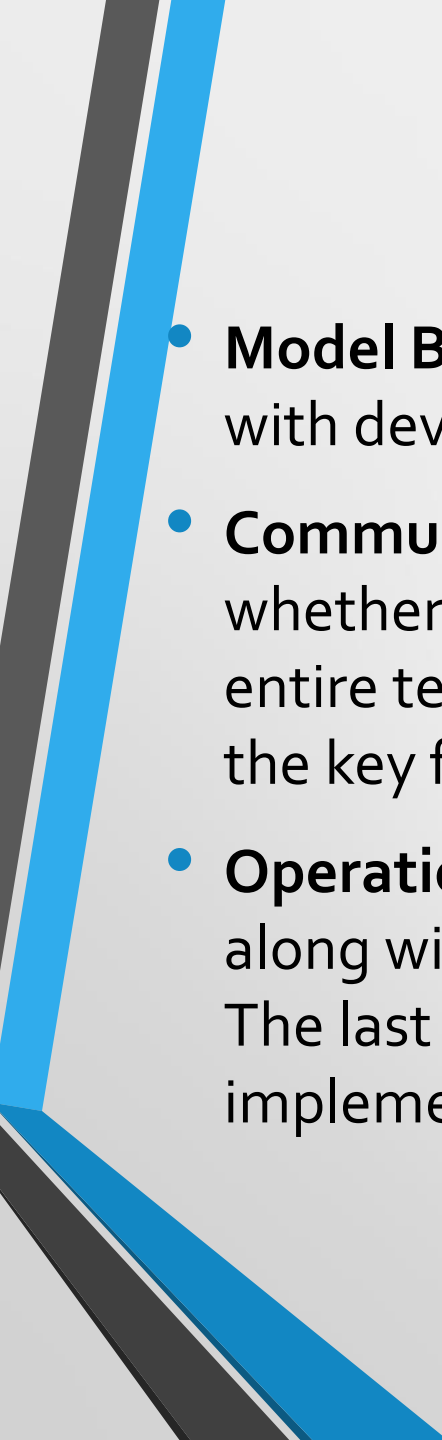
Big Data Analytics Process



All the stakeholders including data analysts, business intelligence analysts, database administrators, data engineers, project managers, and data scientists are involved in the entire process.

It is interesting to note that these six phases of data analytics can follow both forward and backward movement between each phase and are iterative.

- **Data discovery:** This involves examine the business trends, make case studies of similar data analytics, and study the domain of the business industry.
- **Data preparation:** In the second phase after the data discovery phase, data is prepared by transforming it from a legacy system into a data analytics form using different pre-processing techniques.
- **Model planning:** Proper planning of the methods to be adapted and the various workflow to be followed during the next phase of model building. At this stage, the various division of work among the team is decided to clearly define the workload among the team members. The data prepared in the previous phase is further explored to understand the various features and their relationships and also perform feature selection for applying it to the model.

- 
- **Model Building:** This includes execution of the model along with developing datasets for training and testing.
 - **Communicate results :** Checks the results of the project to find whether it is a success or failure. The result is scrutinized by the entire team along with its stakeholders to draw inferences on the key findings and summarize the entire work done..
 - **Operationalization:** A final report is prepared by the team along with the briefings, source codes, and related documents. The last phase also involves running the pilot project to implement the model and test it in a real-time environment.



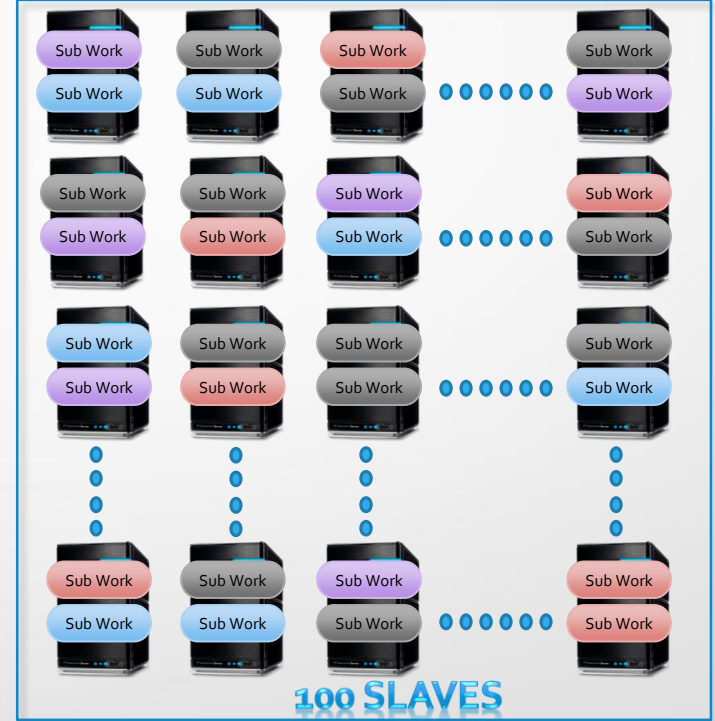
Module 2

What is Hadoop?

The Technology that empowers Yahoo, Facebook, Twitter, Walmart and others



Basic Hadoop Architecture



- Hadoop is a open source framework of storing data in a distributed manner and also doing the processing where the data lies (distributed computing).
- Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.
- Apache top level project, open-source implementation of frameworks that supports data-intensive distributed applications.
- It is a flexible, highly scalable and highly-available architecture for large scale computation and data processing on a network of commodity hardware.
- It facilitate the storage and processing of large and/or rapidly growing data sets including structured and non-structured data.
- A HDFS instance may consist of thousands of server machines, each storing part of the file system's data. Detection of faults and quick, automatic recovery from them is a core architectural goal of HDFS.

Definition by Hortonworks (Vendor of Hadoop)

- Hadoop is an **open source platform** for **distributed storage and distributed processing** of very **large datasets on computer clusters** built from commodity hardware.



What is Hadoop?

- Hadoop is an **open source platform** for distributed storage and distributed processing of very large datasets on computer clusters built from commodity hardware.



Open Source

- ❖ Source code is freely available
- ❖ It may be redistributed and modified

What is Hadoop?

- Hadoop is an open source platform for **distributed storage and distributed processing** of very large datasets on computer clusters built from commodity hardware.



Distributed Processing

- ❖ Data is processed distributedly on multiple nodes / servers
- ❖ Multiple machines processes the data independently

What is Hadoop?

- Hadoop is an open source platform for distributed storage and distributed processing of very large datasets on **computer clusters** built from commodity hardware.



Cluster

- ❖ Multiple machines connected together
- ❖ Nodes are connected via LAN

What is Hadoop?

- Hadoop is an open source platform for distributed storage and distributed processing of very large datasets on computer clusters built from **commodity hardware**.



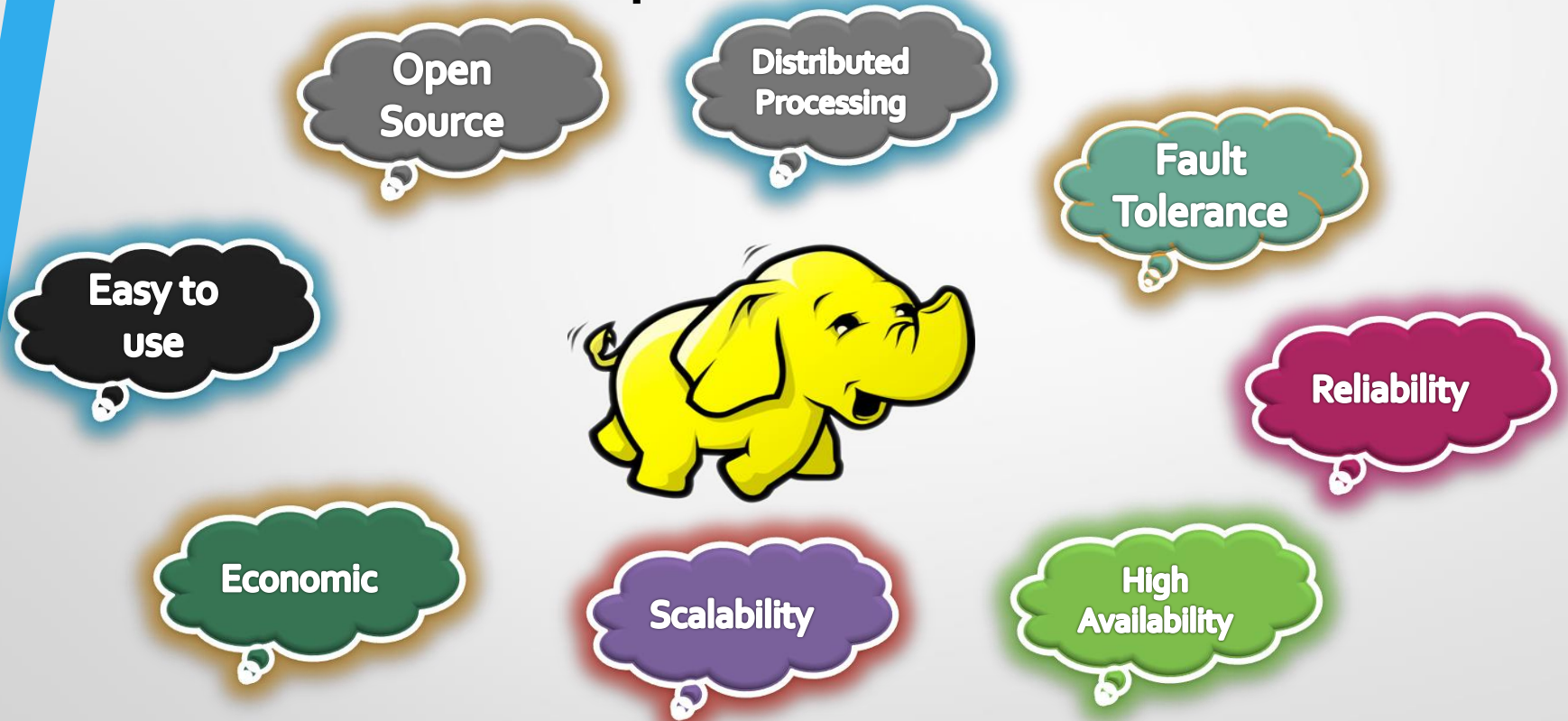
Commodity Hardware

- ❖ Economic / affordable machines
- ❖ Typically low performance hardware

Why use Hadoop?

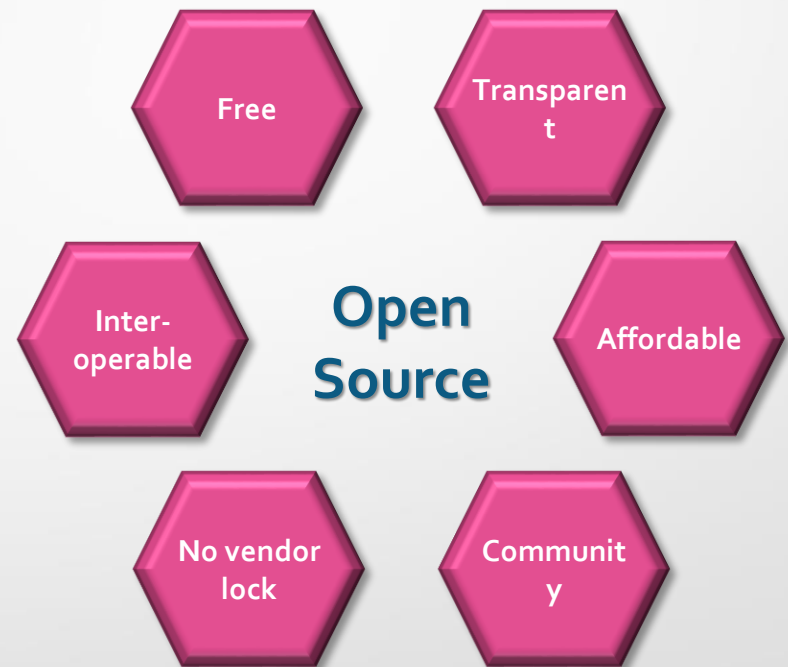
- Need to process Multi Petabyte Datasets
- Data may not have strict schema
- Expensive to build reliability in each application
- Nodes fails everyday
- Need common infrastructure
- Very Large Distributed File System
- Assumes Commodity Hardware
- Optimized for Batch Processing
- Runs on heterogeneous OS

Hadoop Characteristics



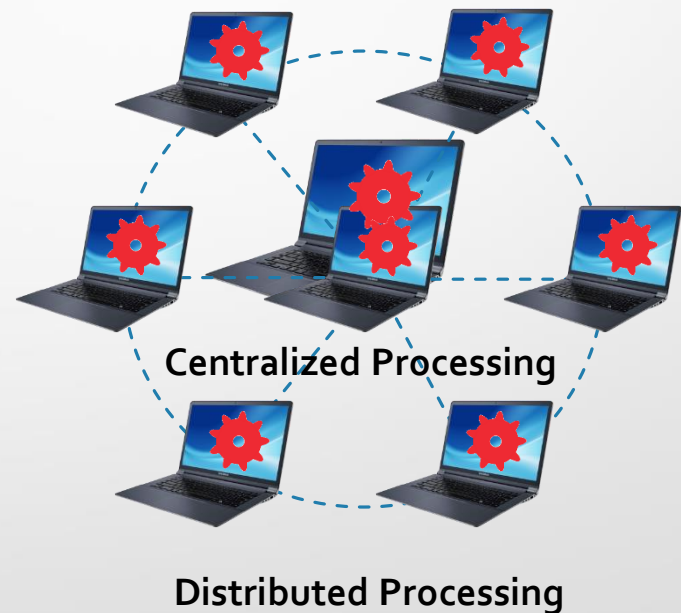
Open Source

- Source code is freely available
- Can be redistributed
- Can be modified



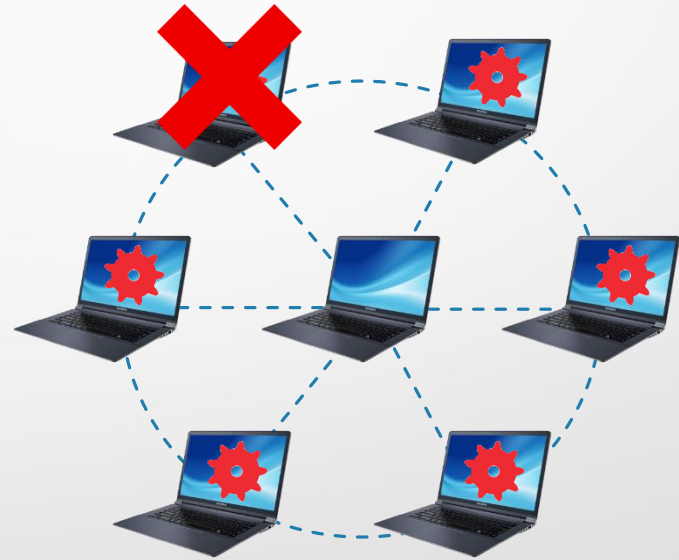
Distributed Processing

- Data is processed distributedly on cluster
- Multiple nodes in the cluster process data independently



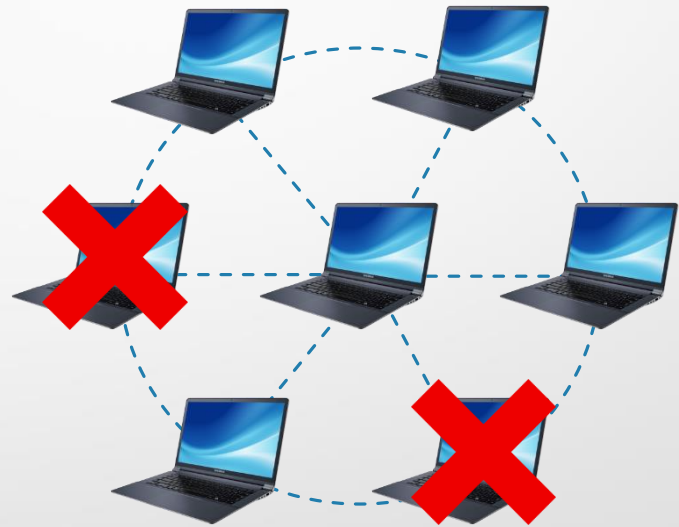
Fault Tolerance

- Failure of nodes are recovered automatically
- Framework takes care of failure of hardware as well tasks



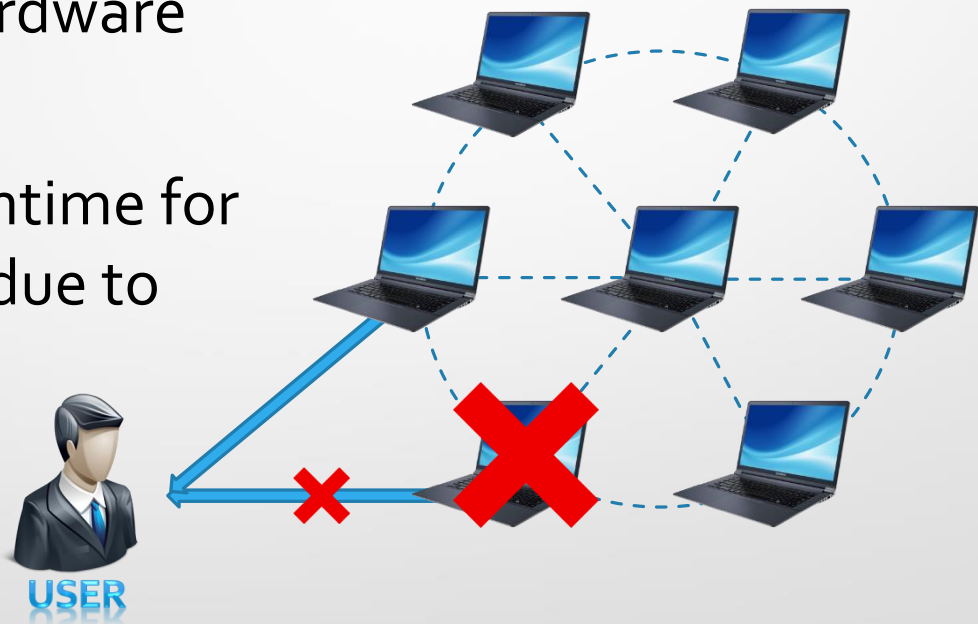
Reliability

- Data is reliably stored on the cluster of machines despite machine failures
- Failure of nodes doesn't cause data loss



High Availability

- Data is highly available and accessible despite hardware failure
- There will be no downtime for end user application due to data



Scalability

- Vertical Scalability – New hardware can be added to the nodes
- Horizontal Scalability – New nodes can be added on the fly



Economic

- No need to purchase costly license
- No need to purchase costly hardware



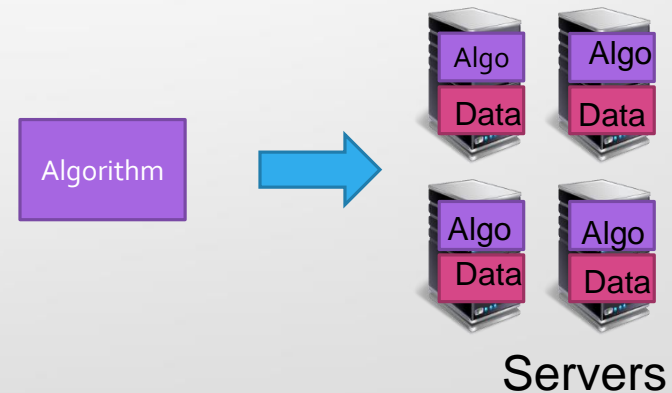
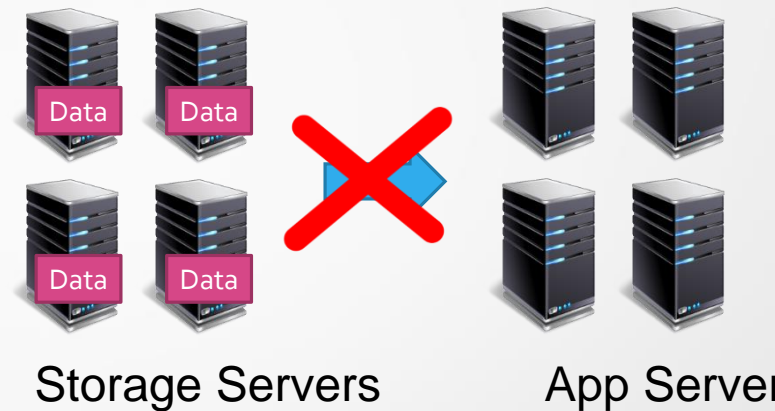
Easy to Use

- Distributed computing challenges are handled by framework
- Client just need to concentrate on business logic



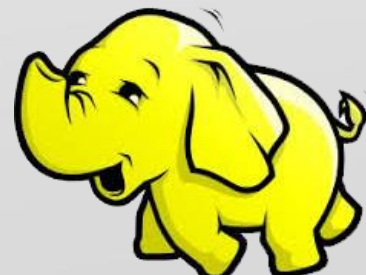
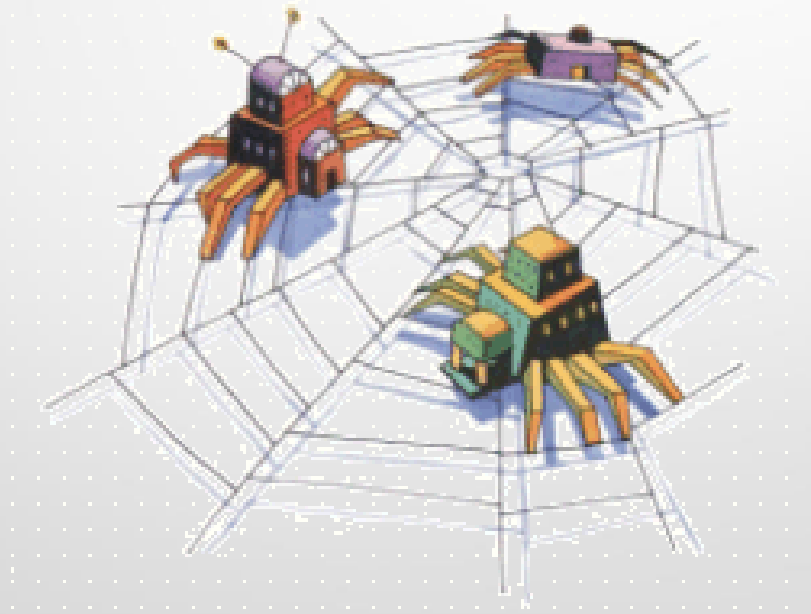
Data Locality

- Move computation to data instead of data to computation
- Data is processed on the nodes where it is stored



Brief History of Hadoop

- Designed to answer the question: **“How to process big data with reasonable cost and time?”**



Hadoop's Developers



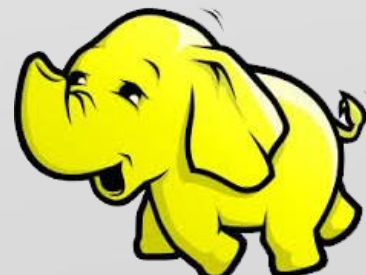
Doug Cutting



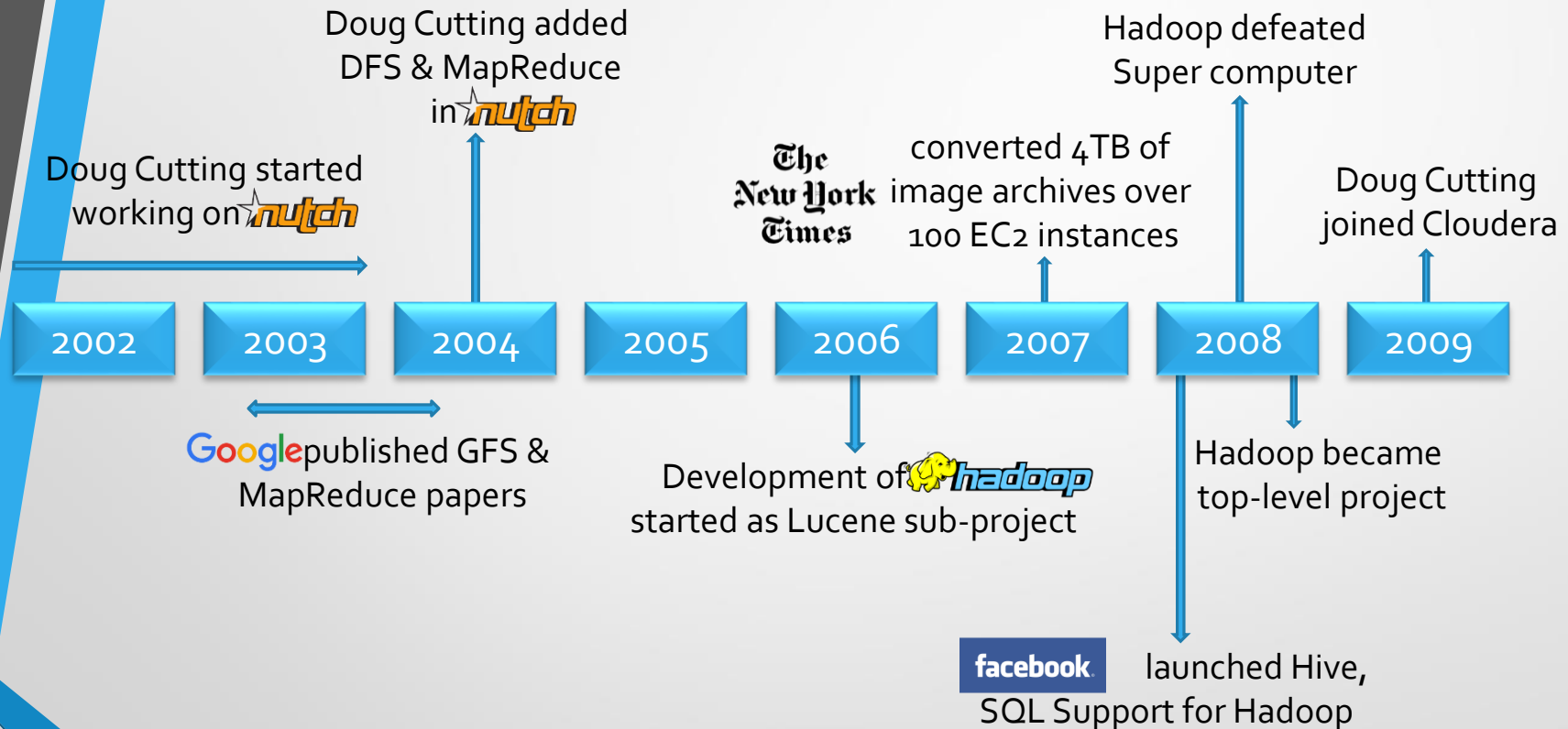
2005: Doug Cutting and Michael J. Cafarella developed Hadoop to support distribution for the [Nutch](#) search engine project.

The project was funded by Yahoo.

2006: Yahoo gave the project to Apache Software Foundation.



Hadoop History



Hadoop Milestones

2003

The Google File System

Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung
Google*



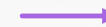
2004

MapReduce: Simplified Data Processing on Large Clusters

Jeffrey Dean and Sanjay Ghemawat

jeff@google.com, sanjay@google.com

Google, Inc.



2006

Bigtable: A Distributed Storage System for Structured Data

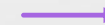
Fuy Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach
Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber
{fuy.jeff.sanjay.wilson.chandra.willach.tushar.fikes.gruber}@google.com

Google, Inc.

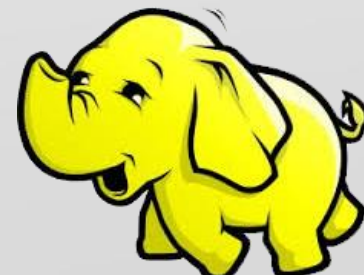
Abstract

Bigtable is a distributed storage system for managing structured data that is designed to scale to a very large number of nodes. It is designed to store and manage petabytes of data across thousands of commodity servers. Many projects at Google store data in Bigtable, including web indexing, Google Earth, and Google Fused Location History. These applications place very different demands on Bigtable, both in terms of data size (from URLs to

achieved scalability and high performance, but Bigtable provides a different interface than such systems. Bigtable does not support a full relational data model; instead, it provides clients with a simple data model that supports dynamic control over data layout and format, and allows clients to reason about the locality properties of data represented in the underlying storage. Data is indexed using row and column names that can be arbitrary strings. Bigtable also treats data as uninterpreted strings.

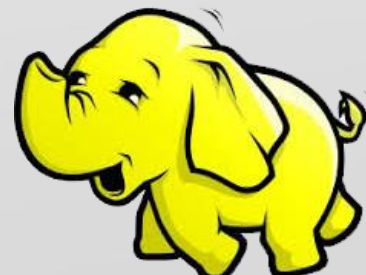


APACHE
HBASE



Hadoop Milestones contd..

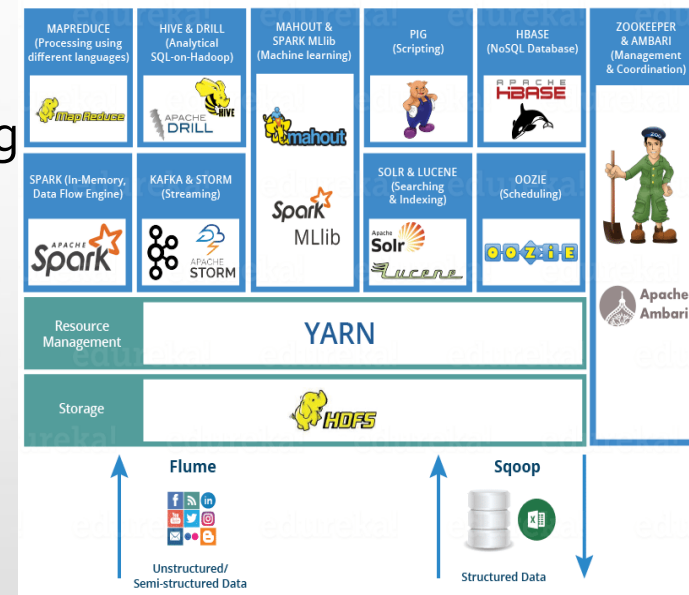
- **2008 - Hadoop Wins Terabyte Sort Benchmark** (sorted 1 terabyte of data in 209 seconds, compared to previous record of 297 seconds)
- 2009 - Avro and Chukwa became new members of Hadoop Framework family
- 2010 - Hadoop's Hbase, Hive and Pig subprojects completed, adding more computational power to Hadoop framework
- **2011 - ZooKeeper Completed**
- **2013 - Hadoop 1.1.2 and Hadoop 2.0.3 alpha.**
 - Ambari, Cassandra, Mahout have been added



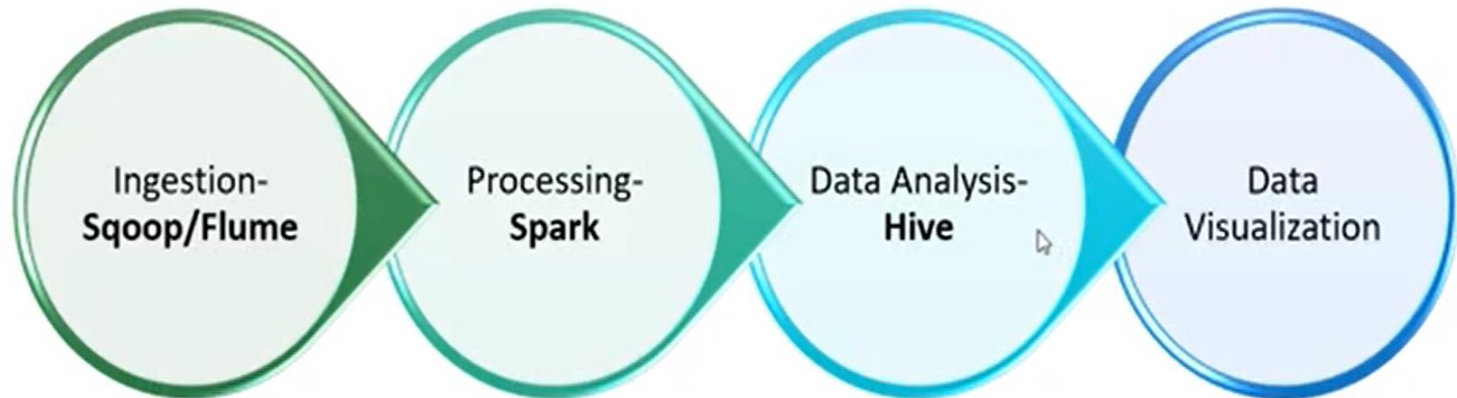
Hadoop Ecosystem

Hadoop components, that together form a Hadoop ecosystem.

- HDFS -> Hadoop Distributed File System (Storage Unit)
- YARN -> Yet Another Resource Negotiator
- MapReduce -> Data processing using programming
- Spark -> In-memory Data Processing
- PIG, HIVE -> Data Processing Services using Query (SQL-like)
- HBase -> NoSQL Database
- Mahout, Spark MLlib -> Machine Learning
- Apache Drill -> SQL on Hadoop
- Zookeeper -> Managing Cluster
- Oozie -> Job Scheduling
- Flume, Sqoop -> Data Ingesting Services
- Solr & Lucene -> Searching & Indexing
- Ambari -> Monitor and Maintain cluster



Big Data Pipelines

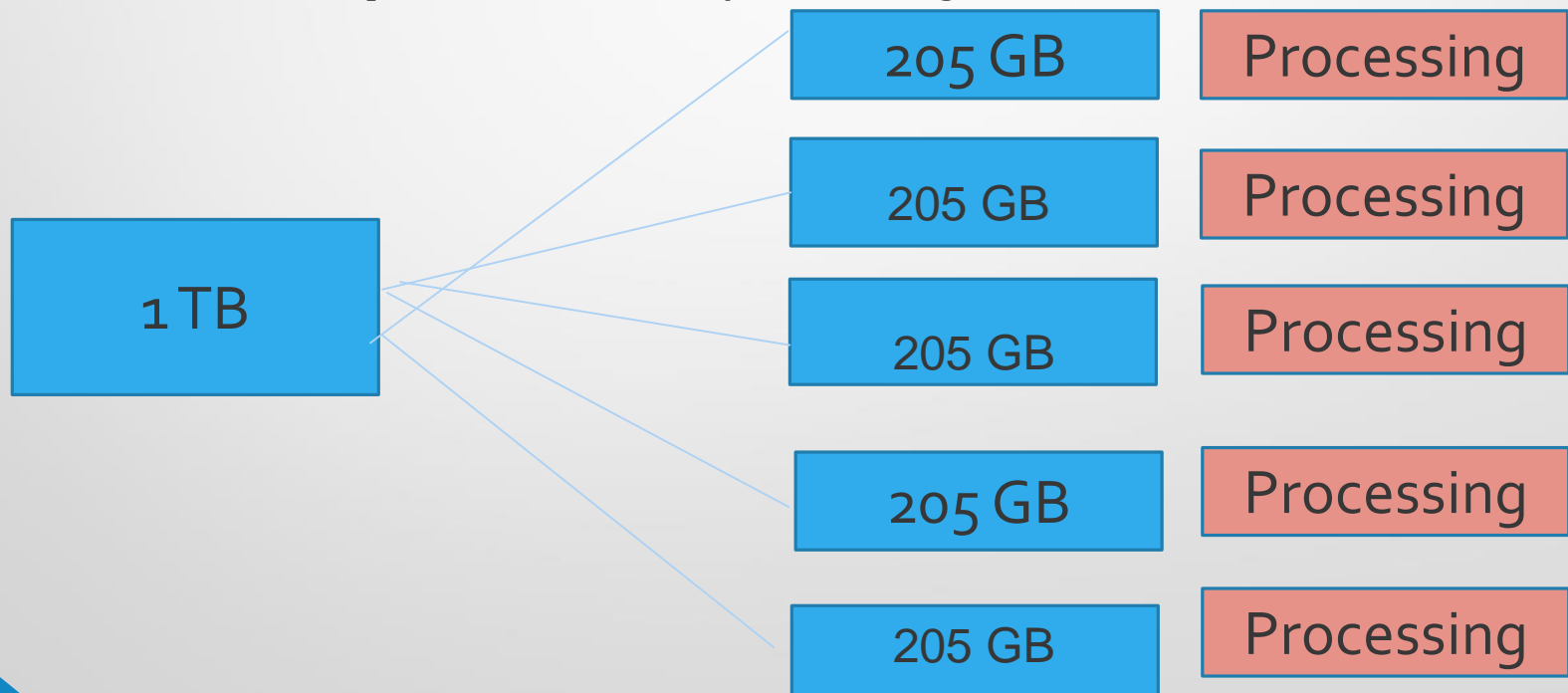


- Bits : 1 or 0
- Bytes: 8 bits – to store a character
- Kilobytes (KB): 1024 Bytes
- Megabytes (MB): 1024 KB
- Gigabytes (GB): 1024 MB
- Terabytes (TB): 1024 GB
- Petabytes (PB): 1024 TB
- Exabytes (EB): 1024 PB
- Zetta Byte (ZB): 1024 EB
- Yotta Byte (YB): 1024 ZB

Big Data Solutions – Two Key Concepts

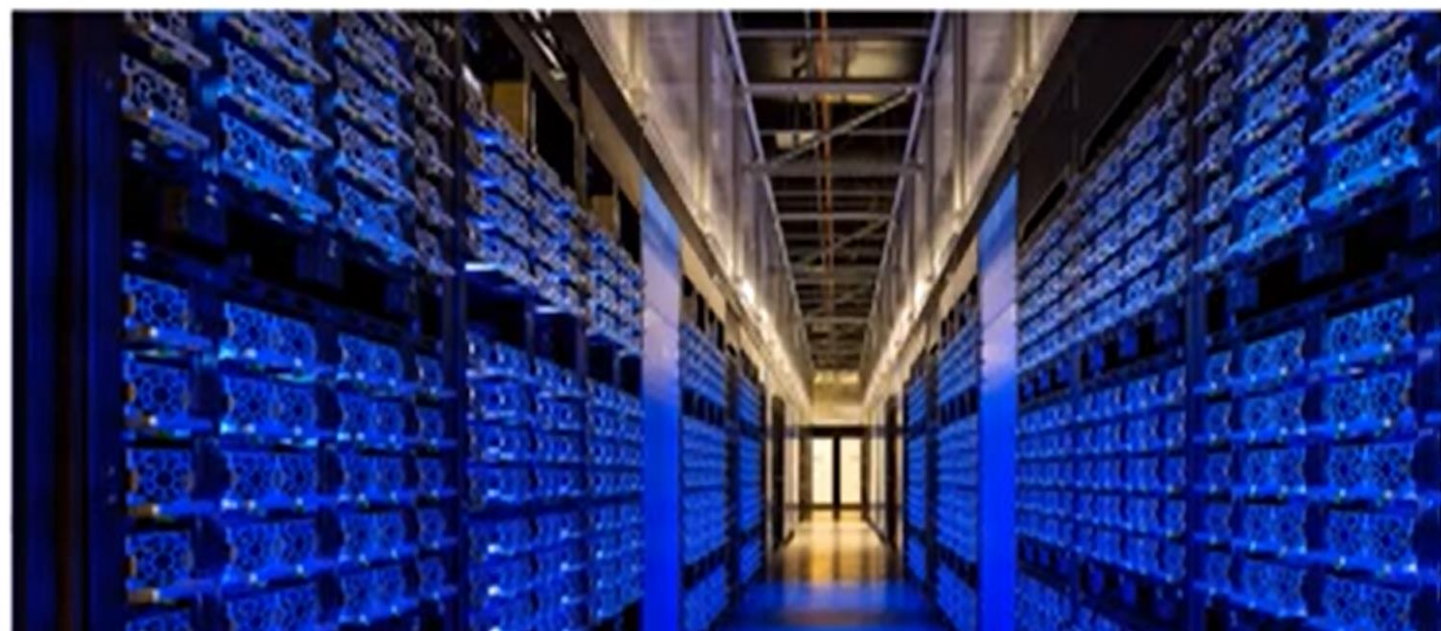
Distributed Storage: The core problem of the big data is the storage problem. This problem is solved using distributed storage. Instead of dumping the entire data at one place, we distribute data into different machines. Since, the hardware is cheap, hence big data storage is affordable. Example: 5 different machines

Distributed computing : Since, the data is stored in distributed manner in different machines, it needs to be processed differently. Processing is done where the data is stored.

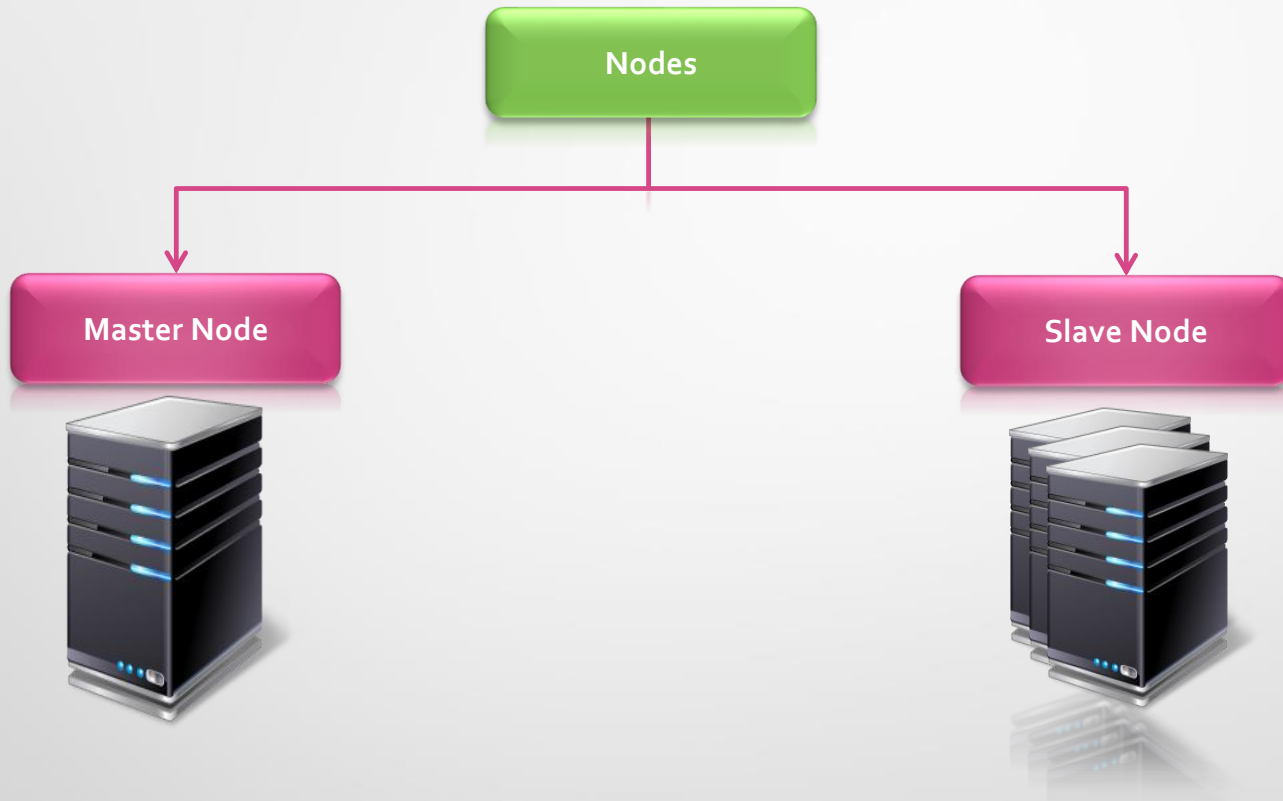


Big data cluster

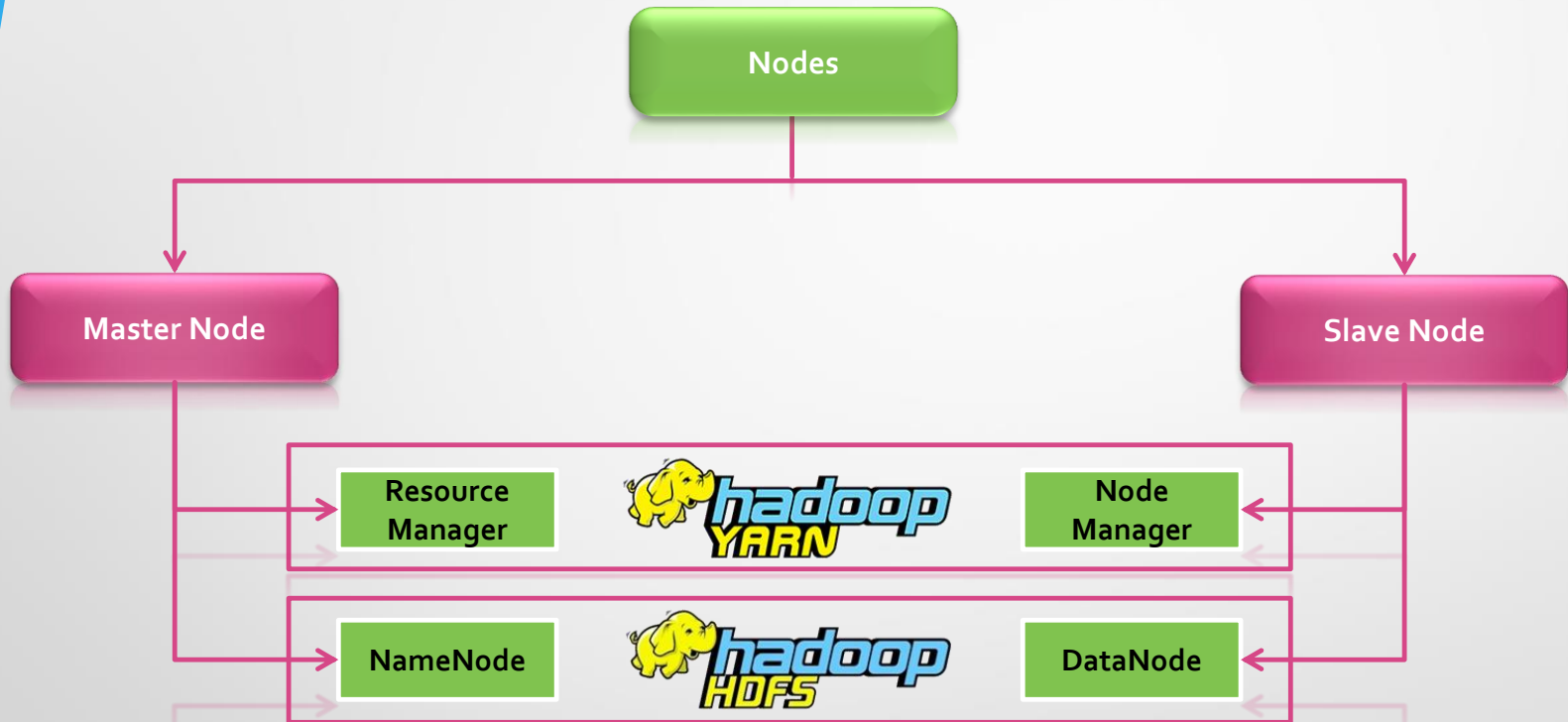
- Machines are connected to each other via network to acts as a single system.
- Machines are nothing but commodity hardware(CPU+RAM)
- These commodity hardware are that are stacked together on a Rack.
- These Racks are then installed in a physical location called as Data Centres.



Hadoop Nodes



Hadoop Daemons



Hadoop Core Components



Hadoop Core Component: HDFS

- HDFS is the core component and the backbone of Hadoop Ecosystem.
- HDFS is the one, which makes it possible to store different types of large data sets (i.e. structured, unstructured and semi structured data).
- - HDFS creates a level of abstraction over the resources, from where we can see the whole HDFS as a single unit. It helps us in storing our data across various nodes and maintaining the log file about the stored data (metadata).
- HDFS is a distributed file system that is fault tolerant, scalable and extremely easy to expand.
- HDFS is the primary distributed storage for Hadoop applications.
- HDFS provides interfaces for applications to move themselves closer to data.
- HDFS is designed to 'just work', however a working knowledge helps in diagnostics and improvements.



Hadoop Core Component: HDFS

HDFS has two core components, i.e. **NameNode and DataNode**. It is a master/slave architecture. HDFS cluster consists of a single Name node, a master server that manages the file system namespace and regulates access to files by clients. HDFS exposes a file system namespace and allows user data to be stored in files.

- Name Node is the heart of an HDFS file system. It maintains and manages the file system metadata, just like a log file or a table of content. E.g; what blocks make up a file, and on which data nodes those blocks are stored, entire metadata is in main memory, List of files, List of Blocks for each file, List of Data Nodes for each block, File attributes, e.g. creation time, replication factor, A Transaction Log containing records file creations, file deletions etc.
- Data Node stores all the actual data and hence it requires more storage resources. A file is split into one or more blocks and set of blocks are stored in Data Nodes. There are a number of Data Nodes usually one per node in a cluster. Data Nodes: serves read, write requests, performs block creation, deletion, and replication upon instruction from Name node. The Data Nodes manage storage attached to the nodes that they run on. These Data Nodes are commodity hardware (like your laptops and desktops) in the distributed environment. That's the reason, why Hadoop solutions are very cost effective. The communication is done to the Name Node for writing the data. Then, it internally sends a request to the client to store and replicate data on various Data Nodes.



Unique features of HDFS

A Master/Slave file system which spreads the Hadoop data over a very large cluster of slave data nodes controlled by a single name node. HDFS also has a bunch of unique features that make it ideal for distributed systems:

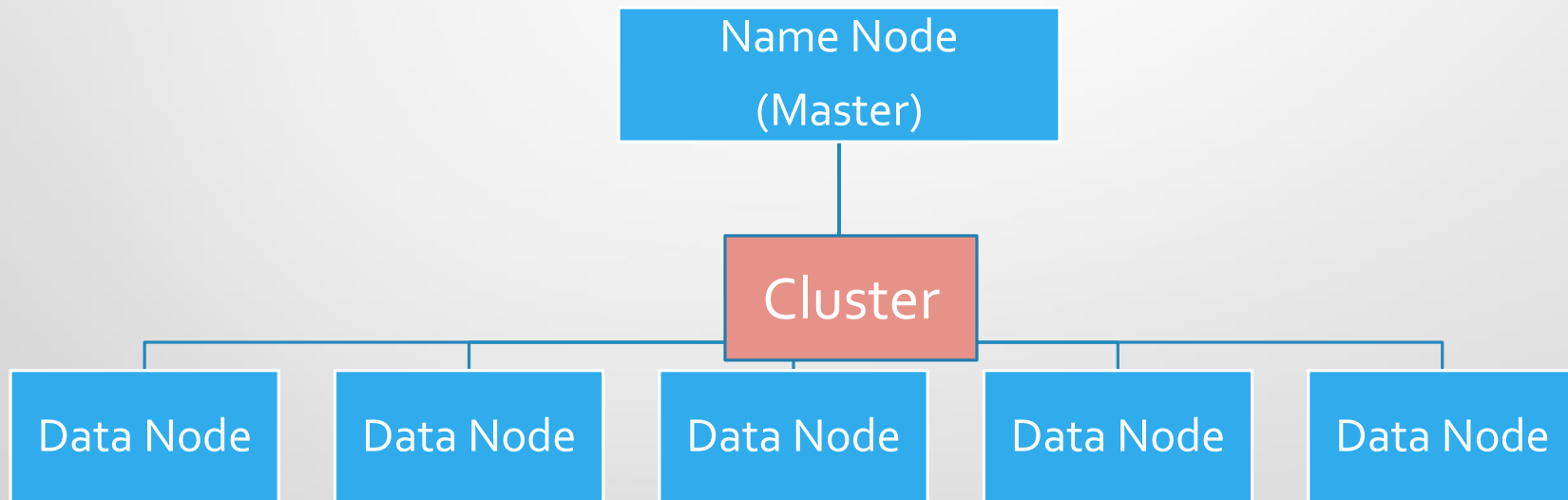
- Failure tolerant - data is duplicated across multiple Data Nodes to protect against machine failures. The default is a replication factor of 3 (every block is stored on three machines).
- Scalability - Data transfers happen directly with the Data Nodes so read/write capacity scales fairly well with the number of Data Nodes
- Space – Helps to add more disk space by just adding more DataNodes.
- Industry standard - Other distributed applications are built on top of HDFS (HBase, Map-Reduce)



HDFS: Scalability

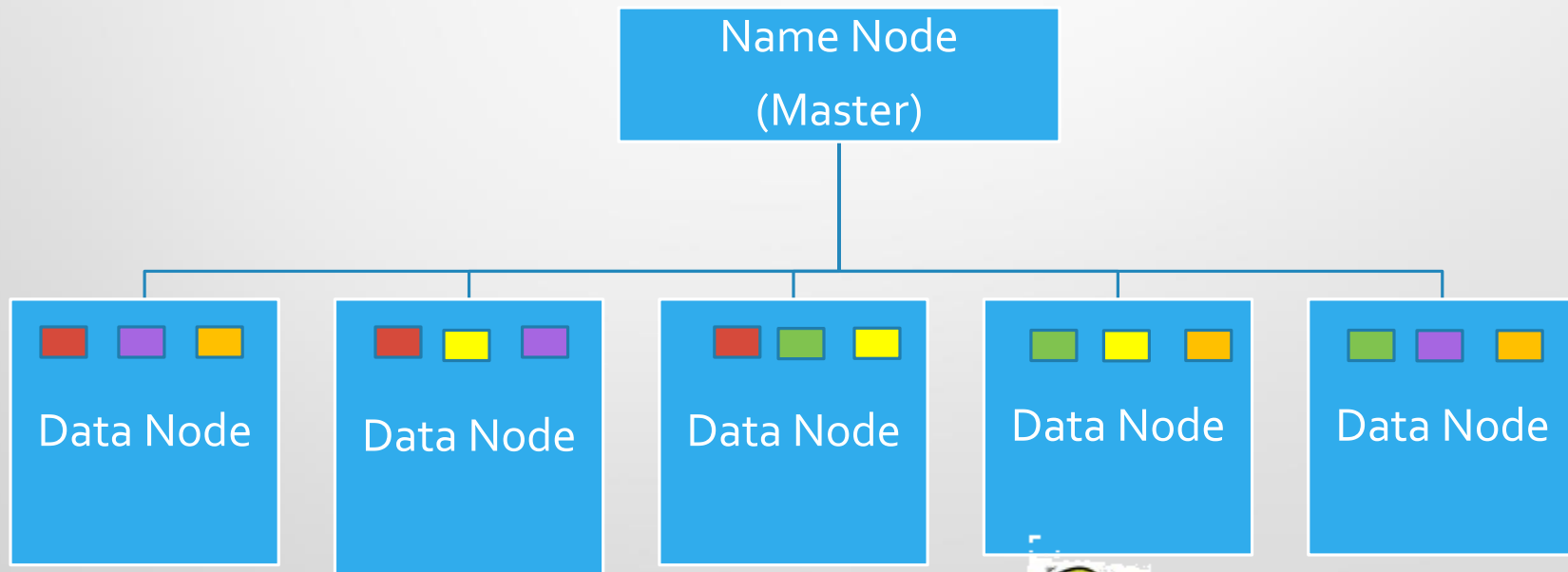
Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage. When Hadoop was first launched, it had two components: HDFS (Distributed Storage) and Map Reduce (Distributed Computing). Multiple machines are assembled to form a cluster of computers. Each machine that stores data is called as data node. Each data node has its own CPU and storage for processing.

The master node manages all the data nodes. It stores meta information. It checks which node is free, which has memory storage and allocate the resources accordingly. Name node act as a central authority in Hadoop framework. It is known as Master-slave architecture. The master node instructs the data node (slave) what to do. The data node stores the information.



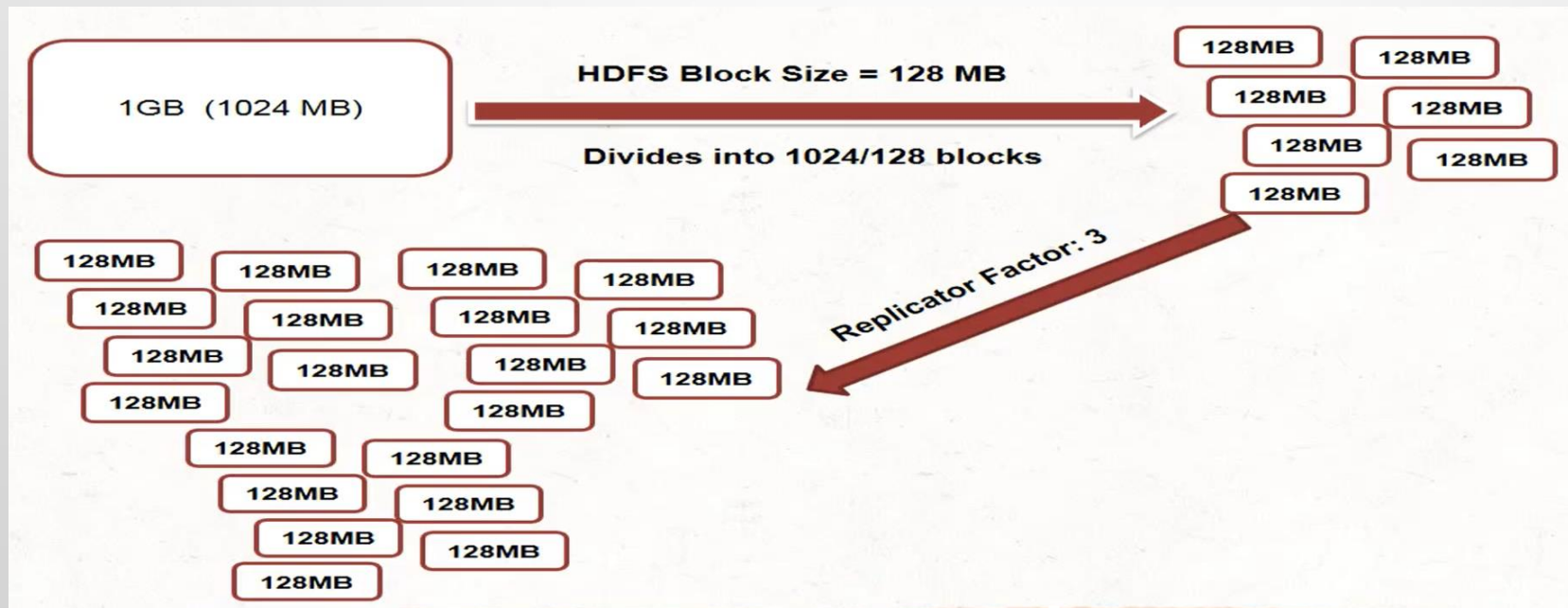
HDFS: Fault Tolerant

HDFS distributes a large dataset among multiple machines (cluster) with each machine having its own CPU and RAM. In HDFS, data is stored in block size of 128 MB and each block is replicated 3 times (default and configurable). The data is still possible to fetch, if data goes down for 2 machines (Fault Tolerance capability). It is possible to increase the replication factor, however that will increase the storage cost. There should be a tradeoff between storage cost and replication factor and is dependent on importance of data.



HDFS: Fault Tolerant

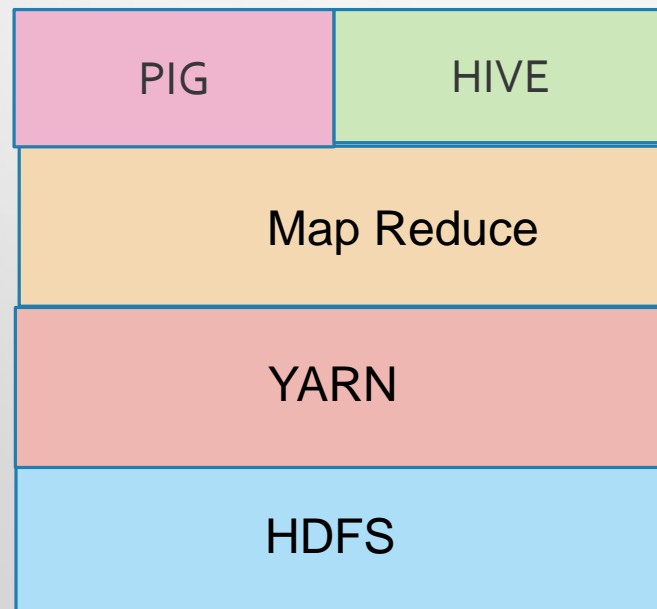
When data becomes large enough to accommodate on a single machine, it becomes necessary to break the data and distribute on multiple machines. This gave birth to HDFS. This is primary data storage. It is distributed means it breaks the file in block of 128 MB and then distributes those blocks on different multiple machines on the cluster. Provide fault tolerance in case of the failures.



Hadoop Core Component: Map Reduce

Initially map reduce programming was done using Java which was very complicated. Later, Pig and Hive query tools were used which have simplified map reduce programming by providing SQL like interface to the end users. Behind the scenes, Pig and Hive queries gets converted to map reduce java programming and get executed on the cluster. The complexity is hidden from the programmers and developers.

In earlier versions of Hadoop, Resource manager was a key component of map reduce and used to do parallel processing and resource management functions. This design resulted in scalability issues and the Hadoop framework became limited only to running map reduce tasks. Later resource management and Job Scheduling was given to YARN in 2012 for relieving MapReduce. Thus, Hadoop earlier had 2 and now has 3 core components: A) HDFS : Distributing large datasets; B) Map Reduce: Performs computation tasks in parallel ; C) YARN: Resource Management and Job Scheduling



Hadoop Core Component: Map Reduce

It is the core component of processing in a Hadoop Ecosystem as it provides the logic of processing. In other words, MapReduce is a software framework which helps in writing applications that processes large data sets using distributed and parallel algorithms inside Hadoop environment. Map reduce consist of several map tasks and reduce tasks. A map reduce job usually splits the input dataset into independent chunks which are processed by the map tasks in a completely parallel manner. In a MapReduce program, **Map()** and **Reduce()** are two functions.

The Map function performs actions like filtering, grouping and sorting.

The map reduce framework sorts the outputs of the maps, which are then input to the reduce tasks. The reduce function aggregates and summarizes the result produced by map function.

The result generated by the Map function is a key value pair (K, V) which acts as the input for Reduce function.

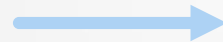
Map Reduce Process

- Each node will have a task tracker.
- Job tracker will send various jobs to task tracker and the task tracker on each node will execute the task on CPU and memory available on each node.
- A client of Hadoop platform will send request to the resource manager to determine average sales and resource manager will send request to node manager to do actual processing.
- The mapper will read the file in the form of key value pair and store the output in disk. It will consider only the relevant fields required for calculating average sales of all states and ignore all the fields which are not required.
- Once the mapper finishes the task , the reducer comes into picture and do the aggregation operations. Example: the reducer will calculate the average sales on each of the nodes. The node manager will send the output from each node to the resource manager and the resource manager will combine the output and send it to the client.
- Thus, map task reads each row and fetches an element. Reduce task performs aggregation operations like sum, count, etc. on the fetched element. Depending on the tasks, reduce job may or may not be required. Example: `select * from customer` will not use reduce jobs.

Map Reduce Example: Calculate average sales for all states from 1 billion records

State	Sales
Maharashtra	2000
Maharashtra	3000
Karnataka	1500
Karnataka	1500
Gujarat	1200

Fetch Sales



Resource Manager
In master node
Job Tracker

Mapper (m)
Reducer (r)
Mapper (m)

Node Manager
Task Tracker

m
r
m

Node Manager
Task Tracker

m
r
m

Node Manager
Task Tracker

m
r
m

Node Manager
Task Tracker

m
r
m

Node Manager
Task Tracker

Determining minimum value from the data using Map – Reduce Algorithm

Data Node

A,24
B,45
C,23
D,10
E,34

Data Node

F,56
G,2
H,22
I,40
J,20

Data Node

K,24
L,-20
M,23
N,10
O,34

Data Node

P,-2
Q,10
R,5
S,0
T,11

Data Node

A,24
B,45
C,23
D,10
E,34

Mapper

Data Node

F,56
G,2
H,22
I,40
J,20

Mapper

Data Node

K,24
L,-20
M,23
N,10
O,34

Mapper

Data Node

P,-2
Q,10
R,5
S,0
T,11

Mapper

Data Node

A,24
B,45
C,23
D,10
E,34

Mapper

D,10

Data Node

F,56
G,2
H,22
I,40
J,20

Mapper

G,2

Reducer

Data Node

K,24
L,-20
M,23
N,10
O,34

Mapper

L,-20

Data Node

P,-2
Q,10
R,5
S,0
T,11

Mapper

P,-2

Data Node

A,24
B,45
C,23
D,10
E,34

Mapper

D,10

Data Node

F,56
G,2
H,22
I,40
J,20

Mapper

G,2

Reducer

L,-20

Data Node

K,24
L,-20
M,23
N,10
O,34

Mapper

L,-20

Data Node

P,-2
Q,10
R,5
S,0
T,11

Mapper

P,-2

Hadoop Core Component: YARN

YARN is the brain of Hadoop Ecosystem. YARN keeps track of storage capacity and compute capacity of all nodes. It performs all your processing activities by allocating resources and scheduling tasks. It has two major components, i.e. Resource Manager and Node Manager.

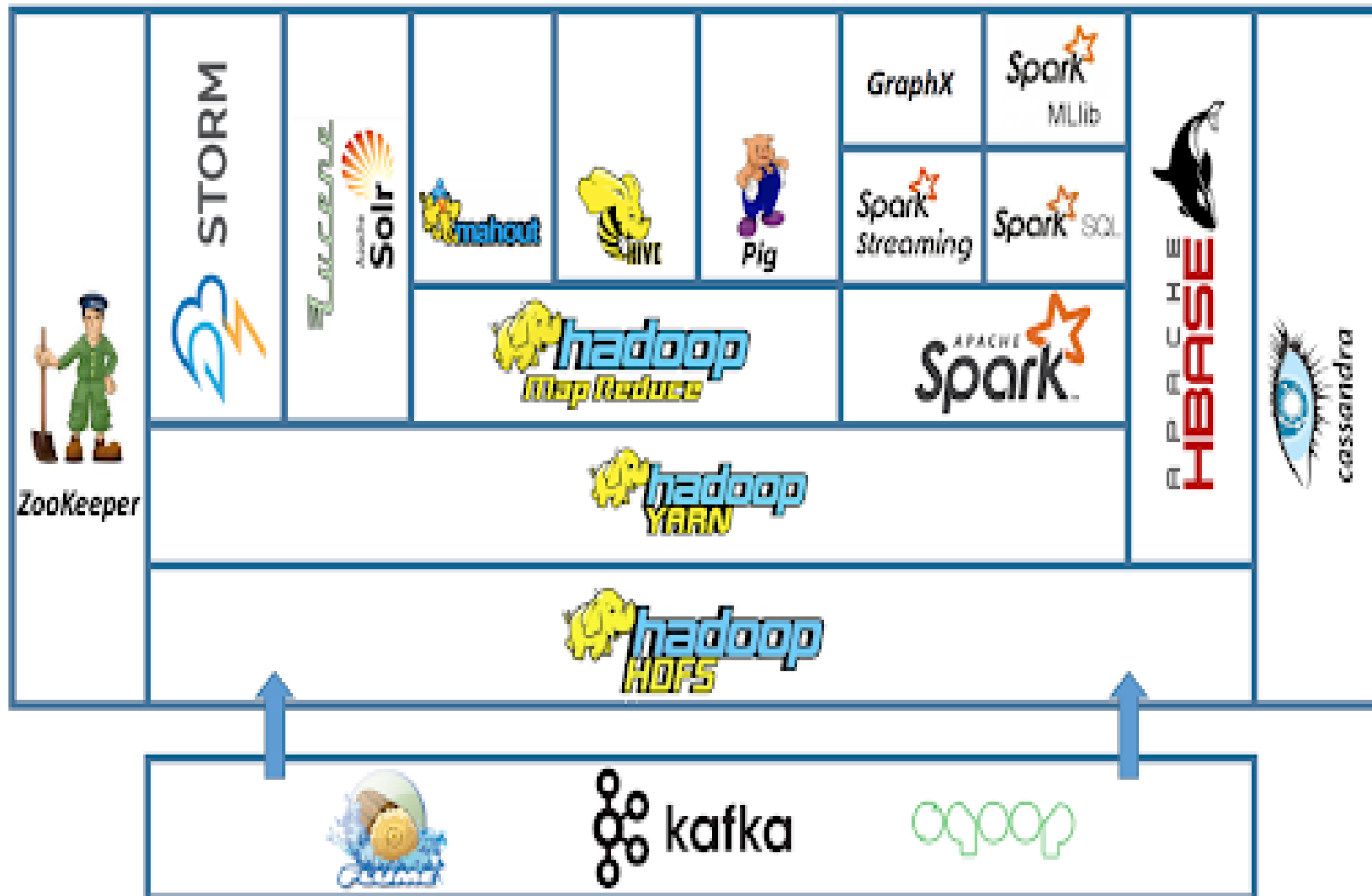
Resource Manager is a main node in the processing department. It receives the processing requests, and then passes the parts of requests to corresponding Node Managers accordingly, where the actual processing takes place. Resource Manager has two components, i.e. Schedulers and Applications Manager. Schedulers perform scheduling algorithms and allocates the resources. While Applications Manager accepts the job submission, negotiates to containers (i.e. the Data node environment where process executes) for executing the application specific Application Master and monitoring the progress. Application Masters are the daemons which reside on Data Node and communicates to containers for execution of tasks on each Data Node.

Node Managers are installed on every Data Node. It is responsible for execution of task on every single Data Node.

Yarn started to give Hadoop an ability to run non Map Reduce Jobs (such as Spark) within the Hadoop framework.



Hadoop Ecosystem



Hadoop Ecosystem: HIVE

Problem: Facebook found it hard to process and analyze big data as not all the employees were well versed with high level coding languages.

Solution: They required a language similar to SQL which was easier to write. Hence, hive was developed with a vision to include the concepts of tables, columns just like SQL. The query language of Hive is called Hive Query Language(HQL), which is very similar like SQL.

At Facebook Hive warehouse contains tens of thousands of tables, stores over 700TB and is used for reporting and ad-hoc analyses by 200 Fb users.



Hadoop Ecosystem: HIVE

Hive is a data warehouse infrastructure tool to process structured data in Hadoop. It is used for data summarization and analysis and for querying of large data systems in the open-source Hadoop platform. HIVE is a SQL like query tool to process the data stored in HDFS.

Map Reduce is very low level and requires customers to write custom programs. HIVE supports queries expressed in SQL-like language called HiveQL which are compiled into Map Reduce jobs that are executed on Hadoop. It converts SQL-like queries into Map Reduce jobs for easy execution and processing of extremely large volumes of data. It resides on top of Hadoop to summarize Big Data, and makes querying and analyzing easy. It is used by different companies. For example, Amazon uses it in Amazon Elastic MapReduce

Hive can process only the structured data. It is not a database and it points to the data stored in HDFS. Hive stores the metadata (data about data) information (schema, columns etc.) of data stored in HDFS, in a relational database called Hive meta store which is outside HDFS.

Hive creates a table which gives a logical view of the data stored in HDFS.

Hadoop Ecosystem: PIG

Problem: Yahoo found it hard to process and analyze big data using map reduce as not all the employees were well versed with complex java codes.

Solution: There was a necessity to process data using a language which was easier than Java. Yahoo researchers developed Pig which was used to process data quickly and easily. Yahoo worked on Pig to facilitate application deployment on Hadoop. Their need mainly was focused on unstructured data

PIG gives you a platform for building data flow for ETL (Extract, Transform and Load), processing and analyzing huge data sets. In PIG, first the load command, loads the data. Then we perform various functions on it like grouping, filtering, joining, sorting, etc. At last, either the data is shown on the screen or store the result back in HDFS.



Difference between HiveQL and Pig Latin

- Hive Query Language is a query language used by Hive to process and analyze data.
 - Declarative language is exactly similar to SQL.
 - HiveQL works on structured data. (If there is a proper structure in JSON, it is called as structured data)
-
- Pig Latin is the procedural data flow language used in Pig to analyze data.
 - Pig Latin is similar to SQL but varies greatly.
 - It is used for structured, semi-structured and unstructured data .
 - 10 lines of Pig Latin code = 200 Lines in Java

Hadoop Ecosystem: HBASE

HBase is an open source, non-relational distributed database. In other words, it is a NoSQL database.

It supports all types of data and that is why, it's capable of handling anything and everything inside a Hadoop ecosystem.

The HBase was designed to run on top of HDFS and provides BigTable like capabilities.

It gives us a fault tolerant way of storing sparse data, which is common in most Big Data use cases.

A P A C H E
HBASE



Hadoop Ecosystem: SPARK

Apache Spark is a **framework for real time data analytics in a distributed computing environment**. The Spark is written in Scala and was originally developed at the University of California, Berkeley. It executes in-memory computations to increase speed of data processing over Map-Reduce.

It is 100x faster than Hadoop for large scale data processing by exploiting in-memory computations and other optimizations. **Therefore, it requires high processing power than Map-Reduce.**

Spark comes packed with high-level libraries, including support for R, SQL, Python, Scala, Java etc.



Hadoop Ecosystem: Apache DRILL

Apache Drill is used to drill into any kind of data. It's an open source application which works with distributed environment to analyze large data sets.

It supports different kinds NoSQL databases and file systems, which is a powerful feature of Drill. So, basically the main aim behind Apache Drill is to provide scalability so that we can process petabytes and exabytes of data efficiently.

The main power of Apache Drill lies **in combining a variety of data stores** just by using a single query. It has a powerful scalability factor in supporting millions of users and serve their query requests over large scale data.



Hadoop Ecosystem: Zookeeper

Apache Zookeeper is the coordinator of any Hadoop job which includes a combination of various services in a Hadoop Ecosystem. Apache Zookeeper coordinates with various services in a distributed environment.

Before Zookeeper, it was very difficult and time consuming to coordinate between different services in Hadoop Ecosystem. The services earlier had many problems with interactions like common configuration while synchronizing data. Even if the services are configured, changes in the configurations of the services make it complex and difficult to handle. The grouping and naming was also a time-consuming factor. Due to the above problems, Zookeeper was introduced. It saves a lot of time by performing synchronization, configuration maintenance, grouping and naming.



Hadoop Ecosystem: Apache Oozie

Apache Oozie act as a clock and alarm service inside Hadoop Ecosystem. For Apache jobs, Oozie has been just like a scheduler. It schedules Hadoop jobs and binds them together as one logical work. There are two kinds of Oozie jobs:

Oozie workflow: These are sequential set of actions to be executed. You can assume it as a relay race. Where each athlete waits for the last one to complete his part.

Oozie Coordinator: These are the Oozie jobs which are triggered when the data is made available to it.



Hadoop Ecosystem: FLUME

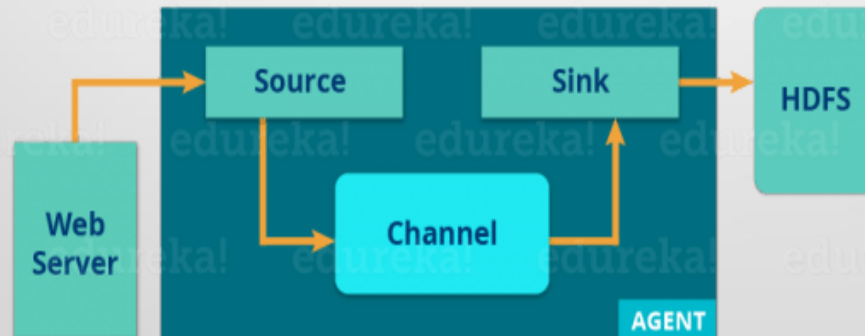
Ingesting data is an important part of our Hadoop Ecosystem. The Flume is a service which helps in ingesting unstructured and semi-structured data into HDFS. It gives us a solution which is reliable and distributed and helps us in collecting, aggregating and moving large amount of data sets. It helps us to ingest online streaming data from various sources like network traffic, social media, email messages, log files etc. in HDFS. Twitter is among one of the famous sources for streaming data.

There is a **Flume agent** which ingests the streaming data from various data sources to HDFS. The flume agent has 3 components: **source, sink and channel**.

1.Source: it accepts the data from the incoming streamline and stores the data in the channel.

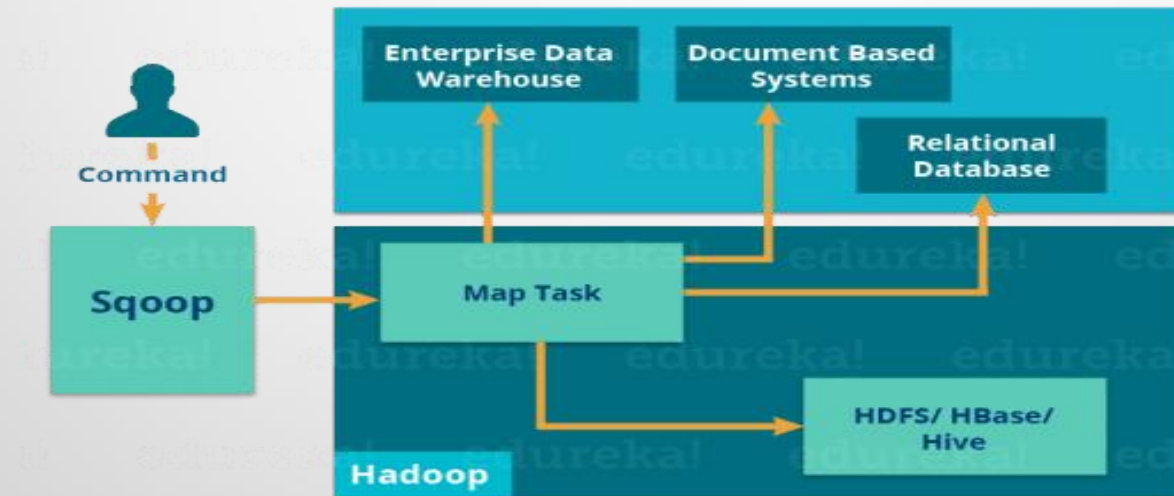
2.Channel: it acts as the local storage or the primary storage. A Channel is a temporary storage between the source of data and persistent data in the HDFS.

3.Sink: Then, our last component i.e. Sink, collects the data from the channel and commits or writes the data in the HDFS permanently.



Hadoop Ecosystem: SQOOP

Flume and Sqoop both are used in ingestion in Hadoop Environment. The major difference between Flume and Sqoop is that Flume only ingests unstructured data or semi-structured data into HDFS. While Sqoop can import as well as export structured data from RDBMS or Enterprise data warehouses to HDFS or vice versa. When Sqoop command is submitted, the main task gets divided into sub tasks which is handled by individual Map Task internally. Map Task is the sub task, which imports part of data to the Hadoop Ecosystem. Collectively, all Map tasks imports the whole data.



Hadoop Ecosystem: Kafka

Apache Kafka is an open-source distributed event streaming platform used by thousands of companies for high-performance data pipelines, streaming analytics, data integration, and mission-critical applications.

*More than **80% of all Fortune 100 companies** trust, and use Kafka*



Hadoop Ecosystem: Solr, Lucene

Apache Solr and Apache Lucene are the two services which are used for searching and indexing in Hadoop Ecosystem.

Apache Lucene is based on Java, which also helps in spell checking.

If Apache Lucene is the engine, Apache Solr is the car built around it. Solr is a complete application built around Lucene. It uses the Lucene Java search library as a core for search and full indexing.

